功能：求n次贝齐尔曲线p(t)由0<=t1<t2<=1界定的子曲线段的贝齐尔点。
输入参数：m_aVertex-控制顶点,为受保护成员，n-次数，顶点数n+1由m_aVertex.GetSize()确定；t1，t2-界定子曲线段的两个参数。
输出参数：m_rVertex-由0<=t1<t2<=1界定的子曲线段的贝齐尔点

```
void AbitBezierDevide(double t1,double t2)
{
        int n= m_aVertex.GetSize();
        if(m_lVertex.GetSize()>0) m_lVertex.RemoveAll();
        if(m_rVertex.GetSize()>0) m_rVertex.RemoveAll();
        m_lVertex.SetSize(n);
        m_rVertex.SetSize(n);
        m_lVertex[0].x=m_aVertex[0].x;
        m_lVertex[0].y=m_aVertex[0].y;
        for(int i=0;i<n;i++)
        {
                m_rVertex[i].x=m_aVertex[i].x;
                m_rVertex[i].y=m_aVertex[i].y;
        }
        if((n-1)!=0)
        {
                for(int i1=1;i1<=n-1;i1++)
                {
                        for(int j=0;j<=n-i1-1;j++)
                        {

m_rVertex[j].x=m_rVertex[j].x+int(t2*double(m_rVertex[j+1].x-m_rVertex[j].x));

m_rVertex[j].y=m_rVertex[j].y+int(t2*double(m_rVertex[j+1].y-m_rVertex[j].y));
                        }
                        m_lVertex[i1]=m_rVertex[0];
                }
                for(int i0=0;i0<n;i0++)
                {
                        m_rVertex[i0].x=m_lVertex[i0].x;
                        m_rVertex[i0].y=m_lVertex[i0].y;
                }
                for(i1=1;i1<=n-1;i1++)
                {
                        for(int j=0;j<=n-i1-1;j++)
                        {

m_rVertex[j].x=m_rVertex[j].x+int(t1/t2*double(m_rVertex[j+1].x-m_rVertex[j].x));

m_rVertex[j].y=m_rVertex[j].y+int(t1/t2*double(m_rVertex[j+1].y-m_rVertex[j].y));
                        }
                }
        }
}
```